

NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMM	MMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNN		NNN	MMMMMM	MMMMMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNNNNN		NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNNNNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL
NNN	NNN	NNN	MMM	MMM	LLLLLLLLLLLLLLLL

_S

Ps

NP

NP

\$G

\$O

NP

PA

_L

```
NN      NN  MM      MM  LL      SSSSSSSS  HH      HH      000000  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  MM      MM  LL      SSSSSSSS  HH      HH      000000  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  MMMM    MMMM LL      SS          HH      HH      00      00  PP      PP  RR      RR  MMMM    MMMM
NN      NN  MMMM    MMMM LL      SS          HH      HH      00      00  PP      PP  RR      RR  MMMM    MMMM
NNNN    NN  MM      MM  LL      SS          HH      HH      00      00  PP      PP  RR      RR  MM      MM
NNNN    NN  MM      MM  LL      SSSSSS      HHHHHHHHHH  00      00  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  MM      MM  LL      SSSSSS      HHHHHHHHHH  00      00  PPPPPPPP  RRRRRRRR  MM      MM
NN      NN  MM      MM  LL      SS          HH      HH      00      00  PP      PP  RR      RR  MM      MM
NN      NN  MM      MM  LL      SS          HH      HH      00      00  PP      PP  RR      RR  MM      MM
NN      NN  MM      MM  LL      SS          HH      HH      00      00  PP      PP  RR      RR  MM      MM
NN      NN  MM      MM  LL      SS          HH      HH      00      00  PP      PP  RR      RR  MM      MM
NN      NN  MM      MM  LLLLLLLLLL SSSSSSSS  HH      HH      000000  PP      PP  RR      RR  MM      MM
NN      NN  MM      MM  LLLLLLLLLL SSSSSSSS  HH      HH      000000  PP      PP  RR      RR  MM      MM
                                                                ....
                                                                ....
                                                                ....
                                                                ....
```

```
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL IIIIII  SSSSSSSS
LLLLLLLLLL IIIIII  SSSSSSSS
```

```

1 0001 0 %TITLE 'NML special volatile parameter handling routines'
2 0002 0 MODULE NML$SHOPRM (
3 0003 0     LANGUAGE (BLISS32),
4 0004 0     ADDRESSING_MODE (NONEXTERNAL=GENERAL),
5 0005 0     ADDRESSING_MODE (EXTERNAL=GENERAL),
6 0006 0     IDENT = 'V04-000'
7 0007 0 ) =
8 0008 1 BEGIN
9 0009 1
10 0010 1 *****
11 0011 1 *
12 0012 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
13 0013 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
14 0014 1 *  ALL RIGHTS RESERVED.
15 0015 1 *
16 0016 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
17 0017 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
18 0018 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
19 0019 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
20 0020 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
21 0021 1 *  TRANSFERRED.
22 0022 1 *
23 0023 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
24 0024 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
25 0025 1 *  CORPORATION.
26 0026 1 *
27 0027 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
28 0028 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
29 0029 1 *
30 0030 1 *
31 0031 1 *****
32 0032 1
33 0033 1
34 0034 1 ++
35 0035 1 FACILITY: DECnet-VAX V2.0 Network Management Listener
36 0036 1
37 0037 1 ABSTRACT:
38 0038 1
39 0039 1     This module contains routines to process volatile data base
40 0040 1     information from the NETACP QIO buffer.
41 0041 1
42 0042 1 ENVIRONMENT: VAX/VMS Operating System
43 0043 1
44 0044 1 AUTHOR: Distributed Systems Software Engineering
45 0045 1
46 0046 1 CREATION DATE: 23-JAN-1980
47 0047 1
48 0048 1 MODIFIED BY:
49 0049 1
50 0050 1     V03-009 MKP0011      Kathy Perko      9-April-1984
51 0051 1     If returning a node address to a Phase III NCP, clear the
52 0052 1     area number if it's in the executor's area. If it's not in
53 0053 1     the executor's area, return it as is - that's the best I
54 0054 1     can do.
55 0055 1
56 0056 1     V03-008 MKP0010      Kathy Perko      18-Oct-1983
57 0057 1     Fix previous bug correctly.

```

58	0058	1	
59	0059	1	
60	0060	1	V03-007 MKP0009 Kathy Perko 27-Sept-1983
61	0061	1	Fix NML\$SHONODEID so it skips over the node name if no
62	0062	1	address is returned.
63	0063	1	
64	0064	1	V03-006 MKP0008 Kathy Perko 17-Aug-1983
65	0065	1	Fix NML\$SHOEXEPARAM to call NML\$SHONODEID for EXECUTOR node
66	0066	1	ALIAS parameter.
67	0067	1	
68	0068	1	V03-005 MKP0007 Kathy Perko 29-July-1983
69	0069	1	Add EXECUTOR node parameter, ALIAS, and clean up routines
70	0070	1	that SHOW node ids.
71	0071	1	
72	0072	1	V03-004 MKP0006 Kathy Perko 29-Nov-1982
73	0073	1	If NCP is using NICE V3.0.0, clear the area number out of
74	0074	1	any node numbers returned.
75	0075	1	
76	0076	1	V03-003 MKP0005 Kathy Perko 24-Nov-1982
77	0077	1	If NETACP doesn't return a state for a node, don't
78	0078	1	return one to NCP.
79	0079	1	
80	0080	1	V03-002 MKP0004 Kathy Perko 25-June-1982
81	0081	1	Executer and X2n Server Destination subaddresses are now
82	0082	1	both returned by the ACP as longwords. Fix up the show
83	0083	1	routines accordingly.
84	0084	1	
85	0085	1	V03-001 MKP0003 Kathy Perko 1-April-1982
86	0086	1	Make changes for X-25 Protocol and Server Modules.
87	0087	1	Also combine some routines to make NMLSHR smaller.
88	0088	1	
89	0089	1	V02-002 MKP0002 Kathy Perko 3-Jan-1982
90	0090	1	Delete routine NML\$SHOLINKS. It has been moved to the
91	0091	1	NMLV2COMP module because it's only used for formatting
92	0092	1	SHOW LINKS commands for V2 nodes.
93	0093	1	
94	0094	1	V02-001 MKP0001 Kathy Perko 24-July-1981
95	0095	1	Delete NML call to map VMS line to DNA line name.
96	0096	1	--

```

98      0097 1 %SBTTL 'Declarations'
99      0098 1
100     0099 1
101     0100 1 TABLE OF CONTENTS:
102     0101 1
103     0102 1
104     0103 1 FORWARD ROUTINE
105     0104 1     NML$SHOPARAM,
106     0105 1     NML$SHONMLVER,
107     0106 1     NML$SHOREMSTA,
108     0107 1     NML$SHOVERSION,
109     0108 1     NML$SHONODEID,
110     0109 1     NML$SHOSERVPA$S,
111     0110 1     NML$SKIPLONG,
112     0111 1     NML$SKIPSTRING,
113     0112 1     NML$SHOEXEPARAM,
114     0113 1     NML$SHORANGE,
115     0114 1     NML$SHOCHANNELS,
116     0115 1     NML$SHOPWSET,
117     0116 1     NML$SHOCOUNTERS,
118     0117 1     NML$SHOOWNER;
119     0118 1
120     0119 1
121     0120 1 INCLUDE FILES:
122     0121 1
123     0122 1
124     0123 1 LIBRARY 'LIB$NMLLIB.L32';
125     0124 1 LIBRARY 'SHRLIB$NMLLIBRY.L32';
126     0125 1 LIBRARY 'SHRLIB$NET.L32';
127     0126 1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
128     0127 1
129     0128 1
130     0129 1 OWN STORAGE:
131     0130 1
132     0131 1
133     0132 1
134     0133 1 Parameter buffer and descriptor for use in handling volatile data base
135     0134 1 data.
136     0135 1
137     0136 1
138     0137 1 OWN
139     0138 1     NML$T_PRMBUFFER : VECTOR [256, BYTE];
140     0139 1 BIND
141     0140 1     NML$Q_PRMDSC = UPLIT (256, NML$T_PRMBUFFER) : DESCRIPTOR;
142     0141 1
143     0142 1
144     0143 1 EXTERNAL REFERENCES:
145     0144 1
146     0145 1
147     0146 1 $NML_EXTDEF;
148     0147 1
149     0148 1 EXTERNAL
150     0149 1     nml$gb_ncp_version,
151     0150 1     nml$gw_vol_exec_addr: BBLOCK [2];
152     0151 1
153     0152 1 EXTERNAL ROUTINE
154     0153 1     NML$ADDMSGCOU,
```

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
Declarations

C 1
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 4
(2)

:	155	0154	1	NML\$ADDMSGPRM,
:	156	0155	1	NML\$LISNMLVER,
:	157	0156	1	NML\$GETNODNAM,
:	158	0157	1	NML\$NETQIO,
:	159	0158	1	NML\$ERROR_1;

```
161 0159 1 GLOBAL ROUTINE NML$SHOPARAM (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
162 0160 1
163 0161 1 !++
164 0162 1 FUNCTIONAL DESCRIPTION:
165 0163 1
166 0164 1 This routine is used to format byte, word, longword, and string NICE
167 0165 1 parameters for SHOW commands. It gets a longword or string parameter
168 0166 1 from the QIO buffer, and adds it to the NICE response message.
169 0167 1
170 0168 1 FORMAL PARAMETERS:
171 0169 1
172 0170 1 SEM_LIST Parameter semantic table entry address.
173 0171 1 BUFDSC Output message buffer descriptor address.
174 0172 1 MSGSIZE Address of current output message size.
175 0173 1 DATDSC QIO buffer descriptor address.
176 0174 1 DATPTR Current pointer into QIO data buffer.
177 0175 1
178 0176 1 ROUTINE VALUE:
179 0177 1 COMPLETION CODES:
180 0178 1
181 0179 1 Always returns success (NML$_STS_SUC).
182 0180 1
183 0181 1 --
184 0182 1
185 0183 2 BEGIN
186 0184 2
187 0185 2 MAP
188 0186 2 SEM_LIST : REF BBLOCK;
189 0187 2
190 0188 2 LOCAL
191 0189 2 DATA_TYPE: BBLOCK [1], ! NICE parameter data type
192 0190 2 NICE_LEN, ! Length of parameter in NICE response message.
193 0191 2 CHECKR_STRING;
194 0192 2
195 0193 2
196 0194 2 Using the NICE data type field in the Parameter Semantic Table (PST),
197 0195 2 determine how long the parameter will be in the NICE response message.
198 0196 2
199 0197 2 CHECK_STRING = 0;
200 0198 2 DATA_TYPE = .SEM_LIST [PST$B_DATATYPE];
201 0199 2
202 0200 2 Check to see if the parameter is coded.
203 0201 2
204 0202 2 IF .DATA_TYPE [NMA$V_PTY_COD] THEN
205 0203 2 BEGIN
206 0204 2 IF .DATA_TYPE [NMA$V_PTY_CMU] THEN
207 0205 2 NML$ERROR_1 (NMA$C_STS_MPR) ! Signal NML error.
208 0206 2 ELSE
209 0207 2
210 0208 2 The parameter is a coded single field. Get the parameter's length
211 0209 2 from the low order 6 bits.
212 0210 2
213 0211 2 NICE_LEN = .DATA_TYPE [NMA$V_PTY_CLE];
214 0212 2 END
215 0213 2 ELSE
216 0214 2
217 0215 2 The parameter is not coded.
```

```
218 0216 2 !
219 0217 3 BEGIN
220 0218 3 IF .DATA_TYPE [NMA$V_PTY_ASC] OR ! NICE parameter type = string
221 0219 3 .DATA_TYPE [NMA$V_PTY_NLE] EQL 0 ! NICE parameter type = binary image
222 0220 3 THEN
223 0221 4 BEGIN
224 0222 4 NICE_LEN = .(.DATPTR) <0,16>;
225 0223 4 .DATPTR = ..DATPTR + 2;
226 0224 4 CHECK_STRING = 1;
227 0225 4 END
228 0226 3 ELSE
229 0227 3 NICE_LEN = .DATA_TYPE [NMA$V_PTY_NLE];
230 0228 2 END;
231 0229 2 !
232 0230 2 If the ACP has a value for the parameter, add it to the NICE response
233 0231 2 message. The ACP does not have a value for the parameter if:
234 0232 2 - It's a string, and the length is zero.
235 0233 2 - It's a longword, and the value is -1.
236 0234 2 The ACP returns only longwords or strings.
237 0235 2 !
238 0236 2 IF (.CHECK_STRING AND .NICE_LEN NEQ 0) OR
239 0237 2 ((NOT .CHECK_STRING) AND (...DATPTR NEQ -1)) THEN
240 0238 2 NML$ADDMSGPRM ( .BUFDSC,
241 0239 2 .MSGSIZE,
242 0240 2 .SEM_LIST [PST$W_DATAID],
243 0241 2 .SEM_LIST [PST$B_DATATYPE],
244 0242 2 .NICE_LEN,
245 0243 2 ..DATPTR);
246 0244 2 !
247 0245 2 !
248 0246 2 Increment the pointer to the QIO P4 buffer to the next parameter
249 0247 2 returned by the ACP.
250 0248 2 !
251 0249 2 IF .CHECK_STRING THEN
252 0250 2 .DATPTR = ..DATPTR + .NICE_LEN
253 0251 2 ELSE
254 0252 2 .DATPTR = ..DATPTR + 4;
255 0253 2 !
256 0254 2 RETURN NML$_STS_SUC
257 0255 2 !
258 0256 1 END; ! End of NML$SHOPARM
```

```
.TITLE NML$SHOPRM NML special volatile parameter handl
ing routine
```

```
.IDENT \V04-000\
```

```
.PSECT $PLITS,NOWRT,NOEXE,2
```

```
00000100 00000 P.AAA:
00000000 00004
```

```
.LONG 256
.ADDRESS NML$_PRMBUFFER
```

```
.PSECT $OWNS,NOEXE,2
```

```
00000 NML$_PRMBUFFER:
```

```
.BLKB 256
```

```
NML$Q_PRMDSC= P.AAA
.EXTRN NML$GB_EVTSRCTYP
.EXTRN NML$GQ_EVTSRCDSC
.EXTRN NML$GW_EVTCLASS
.EXTRN NML$GB_EVTMSKTYP
.EXTRN NML$GQ_EVTMSKDSC
.EXTRN NML$GW_EVTSNKADR
.EXTRN NML$GW_ACP_CHAN
.EXTRN NML$GL_LOGMASK, NML$GQ_ENTSTRDSC
.EXTRN NML$AB_QIOBUFFER
.EXTRN NML$GQ_QIOBFDSC
.EXTRN NML$AB_EXEBUFFER
.EXTRN NML$GL_EXEDATPTR
.EXTRN NML$GQ_EXEDATDSC
.EXTRN NML$GQ_EXEBFDSC
.EXTRN NML$AB_RCVBUFFER
.EXTRN NML$GQ_RCVBFDSC
.EXTRN NML$AB_SNDBUFFER
.EXTRN NML$GQ_SNDBFDSC
.EXTRN NML$GL_RCVDATLEN
.EXTRN NML$AB_CPTABLE, NML$AB_MSGBLOCK
.EXTRN NML$AB_ENTITY_ID
.EXTRN NML$AB_QUALIFIER_ID
.EXTRN NML$AB_ENTITYDATA
.EXTRN NML$AB_NML_NMV, NML$AB_PRMSEM
.EXTRN NML$AB_RECBUF, NML$AL_ENTINFTAB
.EXTRN NML$AL_PERMINFTAB
.EXTRN NML$AW_PRM_DES, NML$GB_CMD_VER
.EXTRN NML$GB_ENTITY_CODE
.EXTRN NML$GB_ENTITY_FORMAT
.EXTRN NML$GL_QUALIFIER_PST
.EXTRN NML$GB_QUALIFIER_FORMAT
.EXTRN NML$GB_FUNCTION
.EXTRN NML$GB_INFO, NML$GB_OPTIONS
.EXTRN NML$GL_PRMCODE, NML$GL_PRS_FLGS
.EXTRN NML$GL_NML_ENTITY
.EXTRN NML$GQ_NETNAMDSC
.EXTRN NML$GQ_RECBFDSC
.EXTRN NML$GW_PRMDSCNT
.EXTRN NML$GB_NCP_VERSION
.EXTRN NML$GW_VOL_EXEC_ADDR
.EXTRN NML$ADDMSGCOU, NML$ADDMSGPRM
.EXTRN NML$LISNMLVER, NML$GETNODNAM
.EXTRN NML$NETQIO, NML$ERROR_1
```

```
.PSECT $CODE$,NOWRT,2
```

			003C 00000	.ENTRY NML\$SHOPARAM, Save R2,R3,R4,R5	: 0159
			55 D4 00002	CLRL CHECK STRING	: 0197
	53	04	AC D0 00004	MOVL SEM_LIST, R3	: 0198
	52	03	A3 90 00008	MOVB 3(R3), DATA_TYPE	: 0202
			17 18 0000C	BGEQ 2\$: 0204
0C	52		06 E1 0000E	BBC #6, DATA_TYPE, 1\$: 0205
	7E		05 CE 00012	MNEGL #5, -(SP)	: 0211
	00		01 FB 00015	CALLS #1, NML\$ERROR_1	
			25 11 0001C	BRB 5\$	
54	52	06	00 EF 0001E 1\$:	EXTZV #0, #6, DATA_TYPE, NICE_LEN	: 0211

05	52		1E 11 00023	BRB	5\$:	0202
	0F		06 E0 00025 2\$:	BBS	#6, DATA_TYPE, 3\$:	0218
			52 93 00029	BITB	DATA_TYPE, #15	:	0219
			10 12 0002C	BNEQ	4\$:	
	50	14	AC D0 0002E 3\$:	MOVL	DATPTR, R0	:	0222
	54	00	B0 3C 00032	MOVZWL	@0(R0), NICE_LEN	:	
	60		02 C0 00036	ADDL2	#2, (R0)	:	0223
	55		01 D0 00039	MOVL	#1, CHECK_STRING	:	0224
			05 11 0003C	BRB	5\$:	0218
54	52		00 EF 0003E 4\$:	EXTZV	#0, #4, DATA_TYPE, NICE_LEN	:	0227
	07		55 E9 00043 5\$:	BLBC	CHECK_STRING, 6\$:	0236
			54 D5 00046	TSTL	NICE_LEN	:	
			10 12 00048	BNEQ	7\$:	
	27		55 E8 0004A	BLBS	CHECK_STRING, 9\$:	0237
	50	14	BC D0 0004D 6\$:	MOVL	@DATPTR, R0	:	
FFFFFFFF	8F		60 D1 00051	CMPL	(R0), #-1	:	
			17 13 00058	BEQL	8\$:	
		14	BC DD 0005A 7\$:	PUSHL	@DATPTR	:	0243
			54 DD 0005D	PUSHL	NICE_LEN	:	0242
	7E	03	A3 9A 0005F	MOVZBL	3(R3), -(SP)	:	0241
	7E		63 3C 00063	MOVZWL	(R3), -(SP)	:	0240
	7E	08	AC 7D 00066	MOVQ	BUFDSC, -(SP)	:	0238
00000000G	00		06 FB 0006A	CALLS	#6, NML\$ADDMSGPRM	:	
	06		55 E9 00071 8\$:	BLBC	CHECK_STRING, 10\$:	0250
	14		54 C0 00074 9\$:	ADDL2	NICE_LEN, @DATPTR	:	
			04 11 00078	BRB	11\$:	
	14		04 C0 0007A 10\$:	ADDL2	#4, @DATPTR	:	0252
	50		01 D0 0007E 11\$:	MOVL	#1, R0	:	0254
			04 00081	RET		:	0256

; Routine Size: 130 bytes, Routine Base: \$CODE\$ + 0000

```
: 260 0257 1 %SBTTL 'NML$SHONMLVER Get NML version number'
: 261 0258 1 GLOBAL ROUTINE NML$SHONMLVER (SEM_TABLE, BUFDSC, MSGSIZE, DUMDSC, DATPTR) =
: 262 0259 1
: 263 0260 1 ++
: 264 0261 1 FUNCTIONAL DESCRIPTION:
: 265 0262 1
: 266 0263 1 This routine moves the network management version number into
: 267 0264 1 the output message as a coded multiple parameter.
: 268 0265 1
: 269 0266 1 FORMAL PARAMETERS:
: 270 0267 1
: 271 0268 1 SEM_TABLE Parameter semantic table entry address.
: 272 0269 1 BUFDSC Output message buffer descriptor.
: 273 0270 1 MSGSIZE Address of current output message size.
: 274 0271 1 DUMDSC Not used.
: 275 0272 1 DATPTR Current pointer into QIO data buffer.
: 276 0273 1
: 277 0274 1 IMPLICIT INPUTS:
: 278 0275 1
: 279 0276 1 NONE
: 280 0277 1
: 281 0278 1 IMPLICIT OUTPUTS:
: 282 0279 1
: 283 0280 1 Parameter is added to output message buffer.
: 284 0281 1
: 285 0282 1 ROUTINE VALUE:
: 286 0283 1 COMPLETION CODES:
: 287 0284 1
: 288 0285 1 Always returns success (NML$_STS_SUC).
: 289 0286 1
: 290 0287 1 SIDE EFFECTS:
: 291 0288 1
: 292 0289 1 NONE
: 293 0290 1
: 294 0291 1 --
: 295 0292 1
: 296 0293 2 BEGIN
: 297 0294 2
: 298 0295 2 NML$LISNMLVER (.SEM_TABLE, .BUFDSC, .MSGSIZE, .DUMDSC);
: 299 0296 2
: 300 0297 2 RETURN NML$_STS_SUC
: 301 0298 2
: 302 0299 1 END; ! End of NML$SHONMLVER
```

```
0000 0000
7E 0C AC 7D 00002
7E 04 AC 7D 00006
00000000G 00 04 FB 0000A
50 01 D0 00011
04 00014
```

```
.ENTRY NML$SHONMLVER, Save nothing
MOVQ MSGSIZE, -(SP)
MOVQ SEM_TABLE, -(SP)
CALLS #4, NML$LISNMLVER
MOVL #1, R0
RET
```

```
: 0258
: 0295
:
: 0297
: 0299
```

; Routine Size: 21 bytes, Routine Base: \$CODE\$ + 0082

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHONMLVER Get NML version number

I 1
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NML\$SHOPRM.B32;1

Page 10
(4)

NM
VO

```
304 0300 1 %SBTTL 'NML$SHOVERSION Get coded multiple version number'
305 0301 1 GLOBAL ROUTINE NML$SHOVERSION (SEM_LIST, BUFDSC, MSGSIZE, DUMDSC, DATPTR)=
306 0302 1
307 0303 1 ++
308 0304 1 FUNCTIONAL DESCRIPTION:
309 0305 1
310 0306 1 This parameter moves network facility version numbers into the
311 0307 1 output message buffer as a coded multiple field. Version numbers
312 0308 1 are a string of three bytes.
313 0309 1
314 0310 1 FORMAL PARAMETERS:
315 0311 1
316 0312 1 SEM_LIST Parameter semantic table entry address.
317 0313 1 BUFDSC Output message buffer descriptor address.
318 0314 1 MSGSIZE Address of current output message size.
319 0315 1 DUMDSC Not used.
320 0316 1
321 0317 1 IMPLICIT INPUTS:
322 0318 1
323 0319 1 NONE
324 0320 1
325 0321 1 IMPLICIT OUTPUTS:
326 0322 1
327 0323 1 The output message buffer contains the coded multiple version number.
328 0324 1
329 0325 1 ROUTINE VALUE:
330 0326 1 COMPLETION CODES:
331 0327 1
332 0328 1 Always returns success (NML$_STS_SUC).
333 0329 1
334 0330 1 SIDE EFFECTS:
335 0331 1
336 0332 1 NONE
337 0333 1
338 0334 1 --
339 0335 1
340 0336 2 BEGIN
341 0337 2
342 0338 2 MAP
343 0339 2 SEM_LIST : REF BLOCK [, BYTE];
344 0340 2
345 0341 2 LOCAL
346 0342 2 BUFFER : VECTOR [6, BYTE],
347 0343 2 LEN,
348 0344 2 PTR;
349 0345 2
350 0346 2 Read version parameter.
351 0347 2
352 0348 2 LEN = .(NML$GL_EXEDATPTR)<0,16>;
353 0349 2
354 0350 2 IF .LEN NEQU 3 ! Length must be 3 bytes
355 0351 2 THEN
356 0352 2 RETURN NML$_STS_MPR;
357 0353 2
358 0354 2 NML$GL_EXEDATPTR = .NML$GL_EXEDATPTR + 2;
359 0355 2
360 0356 2 Add version parameter to message.
```

```

: 361      0357 2 !
: 362      0358 22
: 363      0359 22
: 364      0360 22
: 365      0361 22
: 366      0362 22
: 367      0363 22
: 368      0364 22
: 369      0365 22
: 370      0366 22
: 371      0367 22
: 372      0368 22
: 373      0369 22
: 374      0370 22
: 375      0371 22
: 376      0372 22
: 377      0373 22
: 378      0374 22
: 379      0375 1

PTR = CH$PTR (BUFFER);      ! Point to output buffer

INCR I FROM 0 TO 2 DO
  BEGIN
    CH$WCHAR_A (1, PTR);
    CH$WCHAR_A (CH$RCHAR_A (NML$GL_EXEDATPTR) - '0' , PTR);
  END;

NML$ADDMSGPRM (      .BUFDSC,
                    .MSGSIZE,
                    .SEM_LIST [PST$W_DATAID],
                    .SEM_LIST [PST$B_DATATYPE] OR 3,
                    6,
                    BUFFER);

RETURN NML$_STS_SUC

END;      ! End of NML$SHOVERSION
```

54	00000000G	00	001C	00000	.ENTRY	NML\$SHOVERSION, Save R2,R3,R4	0301
5E		08	9E	00002	MOVAB	NML\$GL_EXEDATPTR, R4	
50		64	D0	00009	SUBL2	#8, SP	
50		60	3C	0000C	MOVL	NML\$GL_EXEDATPTR, R0	0348
03		50	D1	0000F	MOVZWL	(R0), [EN	
		04	13	00012	CMPL	LEN, #3	0350
50		0A	CE	00015	BEQL	1\$	
			04	00017	MNEGL	#10, R0	0352
64		02	C0	0001A	RET		
50		6E	9E	0001B	ADDL2	#2, NML\$GL_EXEDATPTR	0354
		52	D4	0001E	MOVAB	BUFFER, PTR	0358
80		01	90	00021	CLRL	I	0360
53		64	D0	00023	MOVB	#1, (PTR)+	0362
51		63	9A	00026	MOVL	NML\$GL_EXEDATPTR, R3	0363
		64	D6	00029	MOVZBL	(R3), R1	
80	ED	51	83	0002C	INCL	NML\$GL_EXEDATPTR	
		02	F3	0002E	SUBB3	#48, RT, (PTR)+	
		5E	DD	00032	AOBLEQ	#2, I, 2\$	0360
		06	DD	00036	PUSHL	SP	0366
50		04	AC	00038	PUSHL	#6	
51		03	A0	0003A	MOVL	SEM_LIST, R0	0369
7E		03	C9	0003E	MOVZBL	3(R0), R1	
		60	3C	00042	RISL3	#3, R1, -(SP)	
		08	AC	00046	MOVZWL	(R0), -(SP)	0368
		06	FB	00049	MOVQ	BUFDSC, -(SP)	0366
	00000000G	01	D0	00054	CALLS	#6, NML\$ADDMSGPRM	
		04	D0	00057	MOVL	#1, R0	0373
					RET		0375

; Routine Size: 88 bytes, Routine Base: \$CODE\$ + 0097

```
381 0376 1 %SBTTL 'NML$SHOREMSTA Get remote node state'
382 0377 1 GLOBAL ROUTINE NML$SHOREMSTA (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
383 0378 1
384 0379 1 ++
385 0380 1 FUNCTIONAL DESCRIPTION:
386 0381 1
387 0382 1 This routine maps remote node status from the internal NETACP
388 0383 1 bit value to the network management state value. The following
389 0384 1 states are possible:
390 0385 1
391 0386 1 reachable (NETACP value = 1, NML value = NMA$C_STATE_REA)
392 0387 1 unreachable (NETACP value = 0, NML value = NMA$C_STATE_UNR)
393 0388 1
394 0389 1 FORMAL PARAMETERS:
395 0390 1
396 0391 1 SEM_LIST Parameter semantic table entry address.
397 0392 1 BUFDSC Output message buffer descriptor address.
398 0393 1 MSGSIZE Address of current output message size.
399 0394 1 DATDSC QIO buffer descriptor address.
400 0395 1 DATPTR Current pointer into QIO data buffer.
401 0396 1
402 0397 1 ROUTINE VALUE:
403 0398 1 COMPLETION CODES:
404 0399 1
405 0400 1 If NETACP did not know the state of the remote node, returns
406 0401 1 NML$_STS_PTY.
407 0402 1
408 0403 1
409 0404 1 --
410 0405 1
411 0406 2 BEGIN
412 0407 2
413 0408 2 MAP
414 0409 2 SEM_LIST : REF BLOCK [, BYTE];
415 0410 2
416 0411 2 LOCAL
417 0412 2 STATE : BYTE;
418 0413 2
419 0414 2 IF .(..DATPTR) EQLU -1 THEN
420 0415 3 BEGIN
421 0416 3 .DATPTR = ..DATPTR + 4;
422 0417 3 RETURN NML$_STS_PTY;
423 0418 3 END;
424 0419 2
425 0420 2 Map bit setting to correct network management value.
426 0421 2
427 0422 3 STATE = ( IF .(..DATPTR)<0,8> THEN
428 0423 3 NMA$C_STATE_REA ! Reachable
429 0424 3 ELSE
430 0425 3 NMA$C_STATE_UNR); ! Unreachable
431 0426 2
432 0427 2 Add state parameter to message.
433 0428 2
434 0429 2 NML$ADDMSGPRM (.BUFDSC,
435 0430 2 .MSGSIZE,
436 0431 2 .SEM_LIST [PST$W_DATAID],
437 0432 2 .SEM_LIST [PST$B_DATATYPE],
```

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHOREMSTA Get remote node state

M 1
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 14
(6)

```
: 438
: 439
: 440
: 441
: 442
: 443
: 444
: 445

0433 2      1
0434 2      STATE);
0435 2
0436 2 .DATPTR = ..DATPTR + 4;
0437 2
0438 2 RETURN NML$_STS_SUC
0439 2
0440 1 END;
```

! End of NML\$SHOREMSTA

			0000 00000	.ENTRY	NML\$SHOREMSTA, Save nothing	: 0377
	5E		04 C2 00002	SUBL2	#4, SP	
	50	14	BC D0 00005	MOVL	@DATPTR, R0	: 0414
FFFFFFFF	8F		60 D1 00009	CMPL	(R0), #-1	
			08 12 00010	BNEQ	1\$	
14	BC		04 C0 00012	ADDL2	#4, @DATPTR	: 0416
	50		0C CE 00016	MNEGL	#12, R0	: 0417
			04 00019	RET		
	50	14	BC D0 0001A 1\$:	MOVL	@DATPTR, R0	: 0422
	05		60 E9 0001E	BLBC	(R0), 2\$	
	50		04 D0 00021	MOVL	#4, R0	
			03 11 00024	BRB	3\$	
	50		05 D0 00026 2\$:	MOVL	#5, R0	
	6E		50 90 00029 3\$:	MOVB	R0, STATE	
			5E DD 0002C	PUSHL	SP	: 0429
			01 DD 0002E	PUSHL	#1	
	50	04	AC D0 00030	MOVL	SEM LIST, R0	: 0432
	7E	03	A0 9A 00034	MOVZBL	3(R0), -(SP)	
	7E		60 3C 00038	MOVZWL	(R0), -(SP)	: 0431
	7E	08	AC 7D 0003B	MOVQ	BUFDSC, -(SP)	: 0429
00000000G	00		06 FB 0003F	CALLS	#6, NML\$ADDMSGPRM	
14	BC		04 C0 00046	ADDL2	#4, @DATPTR	: 0436
	50		01 D0 0004A	MOVL	#1, R0	: 0438
			04 0004D	RET		: 0440

; Routine Size: 78 bytes, Routine Base: \$CODE\$ + 00EF

```
447 0441 1 %SBTTL 'NML$SHONODEID Get adjacent node id'
448 0442 1 GLOBAL ROUTINE NML$SHONODEID (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
449 0443 1
450 0444 1 ++
451 0445 1 FUNCTIONAL DESCRIPTION:
452 0446 1
453 0447 1 This routine adds the id of the remote node (NMA$C_PCLI_ADJ) to
454 0448 1 which a line is connected to the output message buffer as a coded
455 0449 1 multiple field.
456 0450 1
457 0451 1 FORMAL PARAMETERS:
458 0452 1
459 0453 1 SEM_LIST Parameter semantic table entry address.
460 0454 1 BUFDSC Output message buffer descriptor address.
461 0455 1 MSGSIZE Address of current output message size.
462 0456 1 DATDSC QIO buffer descriptor address.
463 0457 1 DATPTR Current pointer into QIO data buffer.
464 0458 1
465 0459 1
466 0460 1 ROUTINE VALUE:
467 0461 1 COMPLETION CODES:
468 0462 1
469 0463 1 Always returns success (NML$_STS_SUC).
470 0464 1
471 0465 1 SIDE EFFECTS:
472 0466 1
473 0467 1 NONE
474 0468 1
475 0469 1 --
476 0470 1
477 0471 2 BEGIN
478 0472 2
479 0473 2 MAP
480 0474 2 sem_list : BLOCK [, BYTE];
481 0475 2
482 0476 2 LOCAL
483 0477 2 cm_count, ! Coded multiple field count
484 0478 2 name_dsc: VECTOR [2], ! Descriptor of node name.
485 0479 2 name_buf: BBLOCK [6], ! Temporary buffer for node name.
486 0480 2 totlen, ! Total length of field
487 0481 2 nodadr,
488 0482 2 ptr;
489 0483 2
490 0484 2
491 0485 2 Get node address from P4 buffer returned by NETACP and increment pointer
492 0486 2 to the next parameter in the buffer.
493 0487 2
494 0488 2 nodadr = ..datptr<0,32>;
495 0489 2 datptr = ..datptr + 4;
496 0490 2
497 0491 2 If address is zero then don't return this parameter. If there is one,
498 0492 2 skip over the node name parameter before returning.
499 0493 2
500 0494 2 IF .nodadr EQLU -1 THEN
501 0495 2 BEGIN
502 0496 2 IF .sem_list [pst$l_nfbid] EQL nfb$c_aji_add OR
503 0497 2 .sem_list [pst$l_nfbid] EQL nfb$c_ndi_nnd OR
```

```
504 0498 3      .sem_list [pst$l_nfbid] EQL nfb$sc lli_pnn THEN
505 0499 3      .datptr = ..datptr + ..(..datptr)<0,16> + 2;
506 0500 3      RETURN nml$sts_pty;
507 0501 3      END;
508 0502 3
509 0503 3      If the NCP I'm talking to is speaking NICE V3.0.0 or less, and the node
510 0504 3      is in the executor's area, clear the area number from the node number.
511 0505 3      The theory is that the Phase III system should see node's in the executor's
512 0506 3      area normally (for a Phase III system), but node's outside the executor's
513 0507 3      area shouldn't be represented as nodes in the executor's area. So those
514 0508 3      will just have funny addresses because the area number will not be properly
515 0509 3      formatted by the Phase III system.
516 0510 3
517 0511 3      IF CH$RCHAR (nml$gb_ncp_version) LEQ 3 THEN
518 0512 3      BEGIN
519 0513 3      MAP
520 0514 3      nodadr: BBLOCK;
521 0515 3
522 0516 3      IF .nml$gw_vol_exec_addr [nma$sv_area] EQL .nodadr [nma$sv_area] THEN
523 0517 3      nodadr [nma$sv_area] = 0;
524 0518 3      END;
525 0519 3
526 0520 3      ptr = nml$st_prmbuffer;
527 0521 3
528 0522 3      Add node address field.
529 0523 3
530 0524 3      CH$WCHAR A (2, ptr);
531 0525 3      ptr = CH$MOVE (2, nodadr, .ptr);
532 0526 3
533 0527 3      Get the maximum number of fields in the coded multiple (some parameters
534 0528 3      are returned as a node number and name, and some are returned as simply
535 0529 3      a node number.
536 0530 3
537 0531 3      cm_count = .sem_list [pst$b_datatype] AND NOT nma$m_pty_cmu;
538 0532 3
539 0533 3      If a node name is ever part of this parameter, add the node name field
540 0534 3      (provided NETACP returned one) to the NICE message.
541 0535 3
542 0536 3      IF .cm_count EQL 2 THEN
543 0537 3      BEGIN
544 0538 3      SELECTONEU .sem_list [pst$l_nfbid] OF
545 0539 3      SET
546 0540 3      [nfb$sc_aji_add,      ! Circuit adjacent node address
547 0541 3      nfb$sc_ndi_nnd,      ! Node next node to destination
548 0542 3      nfb$sc_lli_pnn]:      ! Logical link partner node
549 0543 3      BEGIN
550 0544 3      name_dsc [0] = ..(..datptr)<0,16>;
551 0545 3      .datptr = ..datptr + 2;
552 0546 3      name_dsc [1] = ..datptr;
553 0547 3      .datptr = ..datptr + .name_dsc [0];
554 0548 3      END;
555 0549 3
556 0550 3      [OTHERWISE]:
557 0551 3      BEGIN
558 0552 3      name_dsc [0] = 6;
559 0553 3      name_dsc [1] = name_buf;
560 0554 3      nml$getnodnam (.nodadr, name_dsc, name_dsc [0]);
```

```
561 0555 3      END;
562 0556 3      TES;
563 0557 3      :
564 0558 3      : If a node name was returned by NETACP, add it to the message
565 0559 3      : parameter.
566 0560 3      :
567 0561 3      IF .name_dsc [0] NEQU 0 THEN
568 0562 4      BEGIN
569 0563 4
570 0564 4      CH$WCHAR_A (nma$m_pty_asc, ptr);
571 0565 4      CH$WCHAR_A (.name_dsc [0], ptr);
572 0566 4      ptr = CH$MOVE (.name_dsc [0], .name_dsc [1], .ptr);
573 0567 4
574 0568 4      END
575 0569 3      ELSE
576 0570 3      cm_count = 1;
577 0571 3      END;
578 0572 3
579 0573 2      totlen = .ptr - nml$t_prmbuffer;
580 0574 2
581 0575 2      : Add node id to output message as a coded multiple field.
582 0576 2
583 0577 2      nml$addmsgprm (.bufdsc,
584 0578 2      .msgsize,
585 0579 2      .sem_list [pst$w_dataid],
586 0580 2      nma$m_pty_cmu OR .cm_count,
587 0581 2      .totlen,
588 0582 2      nml$t_prmbuffer);
589 0583 2
590 0584 2      RETURN nml$_sts_suc
591 0585 2
592 0586 1      END;

! End of nml$shonodeid
```

			01FC 00000	.ENTRY	NML\$SHONODEID, Save R2,R3,R4,R5,R6,R7,R8	: 0442
	58	00000000	00 9E 00002	MOVAB	NML\$t_PRMBUFFER, R8	:
	5E		10 C2 00009	SUBL2	#16, SP	:
	51	14	AC D0 0000C	MOVL	DATPTR, R1	: 0488
	52	00	B1 D0 00010	MOVL	@0(R1), NODADR	:
	61		04 C0 00014	ADDL2	#4, (R1)	: 0489
FFFFFFFF	8F		52 D1 00017	CMPL	NODADR, #-1	: 0494
			31 12 0001E	BNEQ	3\$:
	50	04	AC D0 00020	MOVL	SEM_LIST, R0	: 0496
13010010	8F	0C	A0 D1 00024	CMPL	12(R0), #318832656	:
			14 13 0002C	BEQL	1\$:
02010022	8F	0C	A0 D1 0002E	CMPL	12(R0), #33620002	: 0497
			0A 13 00036	BEQL	1\$:
08020043	8F	0C	A0 D1 00038	CMPL	12(R0), #134348867	: 0498
			0B 12 00040	BNEQ	2\$:
	50	00	B1 3C 00042 1\$:	MOVZWL	@0(R1), R0	: 0499
	50		61 C0 00046	ADDL2	(R1), R0	:
	61	02	A0 9E 00049	MOVAB	2(R0), (R1)	:
	50		0C CE 0004D 2\$:	MNEGL	#12, R0	: 0500
			04 00050	RET		:

50	50	00000000G	00	00	91	00051	3\$:	CMPB	NML\$GB_NCP_VERSION, #3	:	0511
			15	1A	00058			BGTRU	4\$:	
			0A	EF	0005A			EXTZV	#10, #6, NODADR, R0	:	0516
			02	ED	0005F			CMPZV	#2, #6, NML\$GW_VOL_EXEC_ADDR+1, R0	:	
			05	12	00068			BNEQ	4\$:	
			8F	AA	0006A			BICW2	#64512, NODADR	:	0517
			68	9E	0006F	4\$:		MOVAB	NML\$T_PRMBUFFER, PTR	:	0520
			02	90	00072			MOVB	#2, (PTR)+	:	0524
			52	B0	00075			MOVW	NODADR, (PTR)+	:	0525
			AC	D0	00078			MOVL	SEM_LIST, R6	:	0531
			00	EF	0007C			EXTZV	#0, #6, 3(R6), CM_COUNT	:	
			57	D1	00082			CMPL	CM_COUNT, #2	:	0536
			5F	12	00085			BNEQ	9\$:	
			A6	D0	00087			MOVL	12(R6), R0	:	0538
			50	D1	0008B			CMPL	R0, #33620002	:	0540
			12	13	00092			BEQL	5\$:	
			50	D1	00094			CMPL	R0, #134348867	:	
			09	13	0009B			BEQL	5\$:	
			50	D1	0009D			CMPL	R0, #318832656	:	
			12	12	000A4			BNEQ	6\$:	
			B1	3C	000A6	5\$:		MOVZWL	@0(R1), NAME_DSC	:	0544
			02	C0	000AB			ADDL2	#2, (R1)	:	0545
			61	D0	000AE			MOVL	(R1), NAME_DSC+4	:	0546
			AE	C0	000B2			ADDL2	NAME_DSC, (R1)	:	0547
			17	11	000B6			BRB	7\$:	0538
			06	D0	000B8	6\$:		MOVL	#6, NAME_DSC	:	0552
			6E	9E	000BC			MOVAB	NAME_BUF, NAME_DSC+4	:	0553
			AE	9F	000C0			PUSHAB	NAME_DSC	:	0554
			AE	9F	000C3			PUSHAB	NAME_DSC	:	
			52	DD	000C6			PUSHL	NODADR	:	
			03	FB	000C8			CALLS	#3, NML\$GETNODNAM	:	
			AE	D0	000CF	7\$:		MOVL	NAME_DSC, R0	:	0561
			0E	13	000D3			BEQL	8\$:	
			8F	90	000D5			MOVB	#64, (PTR)+	:	0564
			50	90	000D9			MOVB	R0, (PTR)+	:	0565
			50	28	000DC			MOVC3	R0, @NAME_DSC+4, (PTR)	:	0566
			03	11	000E1			BRB	9\$:	0561
			01	D0	000E3	8\$:		MOVL	#1, CM_COUNT	:	0570
			68	9E	000E6	9\$:		MOVAB	NML\$T_PRMBUFFER, R0	:	0573
			50	C3	000E9			SUBL3	R0, PTR, TOTLEN	:	
			8F	BB	000ED			PUSHR	#*M<R0,R8>	:	0581
			8F	C9	000F1			BISL3	#192, CM_COUNT, -(SP)	:	0580
			66	3C	000F9			MOVZWL	(R6), -(SP)	:	0579
			AC	7D	000FC			MOVQ	BUFDSC, -(SP)	:	0577
			06	FB	00100			CALLS	#6, NML\$ADDMSGPRM	:	
			01	D0	00107			MOVL	#1, R0	:	0584
			04	0010A				RET		:	0586

; Routine Size: 267 bytes, Routine Base: \$CODE\$ + 013D

```
594 0587 1 %SBTTL 'NML$SHOOBJPRV Get object privilege mask'
595 0588 1 GLOBAL ROUTINE NML$SHOOBJPRV (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
596 0589 1
597 0590 1 ++
598 0591 1 FUNCTIONAL DESCRIPTION:
599 0592 1
600 0593 1 This routine gets the privilege list (NMA$PCOB_PRV) for a network
601 0594 1 object and adds it to the output message buffer.
602 0595 1
603 0596 1 Currently, only the first longword of the privilege mask can be
604 0597 1 set so that is all that is returned.
605 0598 1
606 0599 1 FORMAL PARAMETERS:
607 0600 1
608 0601 1 SEM_LIST Parameter semantic table entry address.
609 0602 1 BUFDSC Output message buffer descriptor address.
610 0603 1 MSGSIZE Address of current output message size.
611 0604 1 DATDSC QIO buffer descriptor address.
612 0605 1 DATPTR Current pointer into QIO data buffer.
613 0606 1
614 0607 1 IMPLICIT INPUTS:
615 0608 1
616 0609 1 NONE
617 0610 1
618 0611 1 IMPLICIT OUTPUTS:
619 0612 1
620 0613 1 The output message buffer contains the object privilege mask.
621 0614 1
622 0615 1 ROUTINE VALUE:
623 0616 1 COMPLETION CODES:
624 0617 1
625 0618 1 Always returns success (NML$_STS_SUC).
626 0619 1
627 0620 1 SIDE EFFECTS:
628 0621 1
629 0622 1 Destroys the contents of NML$_PRMBUFFER.
630 0623 1
631 0624 1 --
632 0625 1
633 0626 2 BEGIN
634 0627 2
635 0628 2 MAP
636 0629 2 SEM_LIST : REF BLOCK [, BYTE];
637 0630 2
638 0631 2 IF ..(..DATPTR)<0,32> NEQU -1
639 0632 2 THEN
640 0633 2 NML$ADDMSGPRM ( .BUFDSC,
641 0634 2 .MSGSIZE,
642 0635 2 .SEM_LIST [PST$W_DATAID],
643 0636 2 .SEM_LIST [PST$B_DATATYPE] OR 4,
644 0637 2 4,
645 0638 2 ..DATPTR);
646 0639 2
647 0640 2 .DATPTR = ..DATPTR + 4;
648 0641 2
649 0642 2 RETURN NML$_STS_SUC
650 0643 2
```

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHO0BJPRV Get object privilege mask

F 2
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 20
(8)

; 651

0644 1 END;

! End of NML\$SHO0BJPRV

			0004	00000
	52	14	AC	D0 00002
FFFFFFFF	8F	00	B2	D1 00006
			1E	13 0000E
			62	DD 00010
			04	DD 00012
	50	04	AC	D0 00014
7E	51	03	A0	9A 00018
	51		04	C9 0001C
	7E		60	3C 00020
	7E	08	AC	7D 00023
00000000G	00		06	FB 00027
	62		04	C0 0002E
	50		01	D0 00031
			04	00034

1\$:

```
.ENTRY NML$SHO0BJPRV, Save R2
MOVL DATPTR, R2
CMPL @0(R2), #-1
BEQL 1$
PUSHL (R2)
PUSHL #4
MOVL SEM LIST, R0
MOVZBL 3(R0), R1
BISL3 #4, R1, -(SP)
MOVZWL (R0), -(SP)
MOVQ BUFDSC, -(SP)
CALLS #6, NML$ADDMSGPRM
ADDL2 #4, (R2)
MOVL #1, R0
RET
```

: 0588
: 0631
: 0638
: 0633
: 0636
: 0635
: 0633
: 0640
: 0642
: 0644

; Routine Size: 53 bytes, Routine Base: \$CODE\$ + 0248

```

653 0645 1 %SBTTL 'NML$SHOSERVPASS Get service password'
654 0646 1 GLOBAL ROUTINE NML$SHOSERVPASS (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
655 0647 1
656 0648 1 ++
657 0649 1 FUNCTIONAL DESCRIPTION:
658 0650 1
659 0651 1 This routine gets the service password (NMA$C_PCNO_SPA) for the
660 0652 1 remote node and adds it to the output message as a hexadecimal
661 0653 1 number.
662 0654 1
663 0655 1 FORMAL PARAMETERS:
664 0656 1
665 0657 1 SEM_LIST Parameter semantic table entry address.
666 0658 1 BUFDSC Output message buffer descriptor address.
667 0659 1 MSGSIZE Address of current output message size.
668 0660 1 DATDSC QIO buffer descriptor address.
669 0661 1 DATPTR Current pointer into QIO data buffer.
670 0662 1
671 0663 1 IMPLICIT INPUTS:
672 0664 1
673 0665 1 NONE
674 0666 1
675 0667 1 IMPLICIT OUTPUTS:
676 0668 1
677 0669 1 The output message buffer contains the hex service password.
678 0670 1
679 0671 1 ROUTINE VALUE:
680 0672 1 COMPLETION CODES:
681 0673 1
682 0674 1 Always returns success (NML$_STS_SUC).
683 0675 1
684 0676 1 SIDE EFFECTS:
685 0677 1
686 0678 1 NONE
687 0679 1
688 0680 1 --
689 0681 1
690 0682 2 BEGIN
691 0683 2
692 0684 2 MAP
693 0685 2 SEM_LIST : REF BLOCK [, BYTE];
694 0686 2
695 0687 2 LOCAL
696 0688 2 PRMSIZE;
697 0689 2
698 0690 2 PRMSIZE = ..DATPTR<0,16>;
699 0691 2 .DATPTR = ..DATPTR + 2;
700 0692 2
701 0693 2 If the length is zero then the parameter is not set.
702 0694 2
703 0695 2 IF .PRMSIZE EQLU 0
704 0696 2 THEN
705 0697 2 RETURN NML$_STS_PTY;
706 0698 2
707 0699 2 Add the parameter to the message.
708 0700 2
709 0701 2 NML$ADDMSGPRM (.BUFDSC,
```

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHOSERVPASS Get service password

H 2
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 22
(9)

```
: 710      0702      2      .MSGSIZE,  
: 711      0703      2      .SEM_LIST [PST$W_DATAID],  
: 712      0704      2      .SEM_LIST [PST$B_DATATYPE] OR .PRMSIZE,  
: 713      0705      2      .PRMSIZE,  
: 714      0706      2      ..DATPTR);  
: 715      0707      2  
: 716      0708      2      .DATPTR = ..DATPTR + .PRMSIZE;  
: 717      0709      2  
: 718      0710      2      RETURN NML$_STS_SUC  
: 719      0711      2  
: 720      0712      1      END;  
                                ! End of NML$SHOSERVPASS
```

			0004	00000	.ENTRY	NML\$SHOSERVPASS, Save R2	: 0646
	50	14	BC	D0 00002	MOVL	@DATPTR, R0	: 0690
	52		60	3C 00006	MOVZWL	(R0), PRMSIZE	
14	BC		02	C0 00009	ADDL2	#2, @DATPTR	: 0691
			52	D5 0000D	TSTL	PRMSIZE	: 0695
			04	12 0000F	BNEQ	1\$	
	50		0C	CE 00011	MNEGL	#12, R0	: 0697
				04 00014	RET		
		14	BC	DD 00015	PUSHL	@DATPTR	: 0706
			52	DD 00018	PUSHL	PRMSIZE	: 0705
	50	04	AC	D0 0001A	MOVL	SEM_LIST, R0	: 0704
	51	03	A0	9A 0001E	MOVZBL	3(R0), R1	
7E	51		52	C9 00022	BISL3	PRMSIZE, R1, -(SP)	
	7E		60	3C 00026	MOVZWL	(R0), -(SP)	: 0703
	7E	08	AC	7D 00029	MOVQ	BUFD\$C, -(SP)	: 0701
00000000G	00		06	FB 0002D	CALLS	#6, NML\$ADDMSGPRM	
14	BC		52	C0 00034	ADDL2	PRMSIZE, @DATPTR	: 0708
	50		01	D0 00038	MOVL	#1, R0	: 0710
			04	0003B	RET		: 0712

; Routine Size: 60 bytes, Routine Base: \$CODE\$ + 027D

```
722 0713 1 %SBTTL 'NML$SHOLINEID Get line id'
723 0714 1 GLOBAL ROUTINE NML$SHOLINEID (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
724 0715 1
725 0716 1 ++
726 0717 1 FUNCTIONAL DESCRIPTION:
727 0718 1
728 0719 1 This routine reads the line id string and converts it from
729 0720 1 VMS format to DNA format and then adds it to the output message.
730 0721 1
731 0722 1 FORMAL PARAMETERS:
732 0723 1
733 0724 1 SEM_LIST Parameter semantic table entry address.
734 0725 1 BUFDSC Output message buffer descriptor address.
735 0726 1 MSGSIZE Address of current output message size.
736 0727 1 DATDSC QIO buffer descriptor address.
737 0728 1 DATPTR Current pointer into QIO data buffer.
738 0729 1
739 0730 1 IMPLICIT INPUTS:
740 0731 1
741 0732 1 NONE
742 0733 1
743 0734 1 IMPLICIT OUTPUTS:
744 0735 1
745 0736 1 The output message contains the DNA line id.
746 0737 1
747 0738 1 ROUTINE VALUE:
748 0739 1 COMPLETION CODES:
749 0740 1
750 0741 1 Always returns success (NML$_STS_SUC).
751 0742 1
752 0743 1 SIDE EFFECTS:
753 0744 1
754 0745 1 NONE
755 0746 1
756 0747 1 --
757 0748 1
758 0749 2 BEGIN
759 0750 2
760 0751 2 MAP
761 0752 2 SEM_LIST : REF BLOCK [, BYTE];
762 0753 2
763 0754 2 LOCAL
764 0755 2 PRMSIZE;
765 0756 2
766 0757 2 PRMSIZE = ..DATPTR<0,16>;
767 0758 2 .DATPTR = ..DATPTR + 2;
768 0759 2
769 0760 2 If the length is zero then the parameter is not set.
770 0761 2
771 0762 2 IF .PRMSIZE EQLU 0
772 0763 2 THEN
773 0764 2 RETURN NML$_STS_PTY;
774 0765 2
775 0766 2 Add the parameter to the message.
776 0767 2
777 0768 2 NML$ADDMSGPRM ( .BUFDSC,
778 0769 2 .MSGSIZE,
```

```
: 779      0770 2      .SEM_LIST [PST$W_DATAID],
: 780      0771 2      .SEM_LIST [PST$B_DATATYPE],
: 781      0772 2      .PRMSIZE,
: 782      0773 2      ..DATPTR);
: 783      0774 2
: 784      0775 2      .DATPTR = ..DATPTR + .PRMSIZE;
: 785      0776 2
: 786      0777 2      RETURN NML$_STS_SUC
: 787      0778 2
: 788      0779 1      END;                                ! End of NML$SHOLINEID
```

			0004 00000	.ENTRY NML\$SHOLINEID, Save R2	: 0714
	50	14	BC D0 00002	MOVL @DATPTR, R0	: 0757
	52		60 3C 00006	MOVZWL (R0), PRMSIZE	
14	BC		02 C0 00009	ADDL2 #2, @DATPTR	: 0758
			52 D5 0000D	TSTL PRMSIZE	: 0762
			04 12 0000F	BNEQ 1\$	
	50		0C CE 00011	MNEGL #12, R0	: 0764
			04 00014	RET	
		14	BC DD 00015 1\$:	PUSHL @DATPTR	: 0773
			52 DD 00018	PUSHL PRMSIZE	: 0772
	50	04	AC D0 0001A	MOVL SEM_LIST, R0	: 0771
	7E	03	A0 9A 0001E	MOVZBL 3(R0), -(SP)	
	7E		60 3C 00022	MOVZWL (R0), -(SP)	: 0770
	7E	08	AC 7D 00025	MOVQ BUFD\$C, -(SP)	: 0768
00000000G	00		06 FB 00029	CALLS #6, NML\$ADDMSGPRM	
14	BC		52 C0 00030	ADDL2 PRMSIZE, @DATPTR	: 0775
	50		01 D0 00034	MOVL #1, R0	: 0777
			04 00037	RET	: 0779

; Routine Size: 56 bytes, Routine Base: \$CODE\$ + 02B9

```

: 790      0780 1 %SBTTL 'NML$SKIPLONG Skip longword in QIO P4 buffer'
: 791      0781 1 GLOBAL ROUTINE NML$SKIPLONG (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
: 792      0782 1
: 793      0783 1 ++
: 794      0784 1 FUNCTIONAL DESCRIPTION:
: 795      0785 1
: 796      0786 1 This routine skips (advances the pointer past) a byte, word, or
: 797      0787 1 longword parameter in the QIO P4 buffer. Note that the ACP always
: 798      0788 1 returns these parameters in a longword.
: 799      0789 1
: 800      0790 1 FORMAL PARAMETERS:
: 801      0791 1
: 802      0792 1 SEM_LIST Parameter semantic table entry address.
: 803      0793 1 BUFDSC Output message buffer descriptor address.
: 804      0794 1 MSGSIZE Address of current output message size.
: 805      0795 1 DATDSC QIO buffer descriptor address.
: 806      0796 1 DATPTR Current pointer into QIO data buffer.
: 807      0797 1
: 808      0798 1 IMPLICIT INPUTS:
: 809      0799 1
: 810      0800 1 NONE
: 811      0801 1
: 812      0802 1 IMPLICIT OUTPUTS:
: 813      0803 1
: 814      0804 1 NONE
: 815      0805 1
: 816      0806 1 ROUTINE VALUE:
: 817      0807 1 COMPLETION CODES:
: 818      0808 1
: 819      0809 1 Always returns success (NML$_STS_SUC).
: 820      0810 1
: 821      0811 1 SIDE EFFECTS:
: 822      0812 1
: 823      0813 1 NONE
: 824      0814 1
: 825      0815 1 --
: 826      0816 1
: 827      0817 2 BEGIN
: 828      0818 2
: 829      0819 2 .DATPTR = ..DATPTR + 4;
: 830      0820 2
: 831      0821 2 RETURN NML$_STS_SUC
: 832      0822 2
: 833      0823 1 END;

```

! End of NML\$SKIPLONG

```

14 BC 0000 00000
04 C0 00002
01 D0 00006
04 00009

```

```

.ENTRY NML$SKIPLONG, Save nothing
ADDL2 #4, @DATPTR
MOVL #1, R0
RET

```

```

: 0781
: 0819
: 0821
: 0823

```

; Routine Size: 10 bytes, Routine Base: \$CODE\$ + 02F1

```
: 835 0824 1 %SBTTL 'NML$SKIPSTRING Skip string parameter'
: 836 0825 1 GLOBAL ROUTINE NML$SKIPSTRING (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
: 837 0826 1
: 838 0827 1 ++
: 839 0828 1 FUNCTIONAL DESCRIPTION:
: 840 0829 1
: 841 0830 1 This routine skips (advances the pointer past) a string parameter
: 842 0831 1 in the QIO buffer.
: 843 0832 1
: 844 0833 1 FORMAL PARAMETERS:
: 845 0834 1
: 846 0835 1 SEM_LIST Parameter semantic table entry address.
: 847 0836 1 BUFDSC Output message buffer descriptor address.
: 848 0837 1 MSGSIZE Address of current output message size.
: 849 0838 1 DATDSC QIO buffer descriptor address.
: 850 0839 1 DATPTR Current pointer into QIO data buffer.
: 851 0840 1
: 852 0841 1 IMPLICIT INPUTS:
: 853 0842 1
: 854 0843 1 NONE
: 855 0844 1
: 856 0845 1 IMPLICIT OUTPUTS:
: 857 0846 1
: 858 0847 1 NONE
: 859 0848 1
: 860 0849 1 ROUTINE VALUE:
: 861 0850 1 COMPLETION CODES:
: 862 0851 1
: 863 0852 1 Always returns success (NML$_STS_SUC).
: 864 0853 1
: 865 0854 1 SIDE EFFECTS:
: 866 0855 1
: 867 0856 1 NONE
: 868 0857 1
: 869 0858 1 --
: 870 0859 1
: 871 0860 2 BEGIN
: 872 0861 2
: 873 0862 2 LOCAL
: 874 0863 2 LEN;
: 875 0864 2
: 876 0865 2 LEN = ..(..DATPTR)<0,16>;
: 877 0866 2 .DATPTR = ..DATPTR + 2;
: 878 0867 2 .DATPTR = ..DATPTR + .LEN;
: 879 0868 2
: 880 0869 2 RETURN NML$_STS_SUC
: 881 0870 2
: 882 0871 1 END; ! End of NML$SKIPSTRING
```

```
50 14 0000 00000
50 BC D0 00002
14 BC 60 3C 00006
02 C0 00009
```

```
.ENTRY NML$SKIPSTRING, Save nothing
MOVL @DATPTR, R0
MOVZWL (R0), LEN
ADDL2 #2, @DATPTR
```

```
: 0825
: 0865
: 0866
```

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SKIPSTRING Skip string parameter

M 2
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NML\$SHOPRM.B32;1

Page 27
(12)

14 BC 50 C0 0000D
50 01 D0 00011
04 00014

ADDL2 LEN, @DATPTR
MOVL #1, R0
RET

: 0867
: 0869
: 0871

; Routine Size: 21 bytes, Routine Base: \$CODE\$ + 02FB

```

: 884      0872 1 %SBTTL 'NML$SHOEXEPARAM Show executor parameter'
: 885      0873 1 GLOBAL ROUTINE NML$SHOEXEPARAM (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
: 886      0874 1
: 887      0875 1 !++
: 888      0876 1 FUNCTIONAL DESCRIPTION:
: 889      0877 1
: 890      0878 1 This routine adds a parameter from the executor data buffer to the
: 891      0879 1 output message.
: 892      0880 1
: 893      0881 1 FORMAL PARAMETERS:
: 894      0882 1
: 895      0883 1 SEM_LIST Parameter semantic table entry address.
: 896      0884 1 BUFDSC Output message buffer descriptor address.
: 897      0885 1 MSGSIZE Address of current output message size.
: 898      0886 1 DATDSC QIO buffer descriptor address.
: 899      0887 1 DATPTR Current pointer into QIO data buffer.
: 900      0888 1
: 901      0889 1 ROUTINE VALUE:
: 902      0890 1 COMPLETION CODES:
: 903      0891 1
: 904      0892 1 Always returns success (NML$_STS_SUC).
: 905      0893 1
: 906      0894 1 !--
: 907      0895 1
: 908      0896 2 BEGIN
: 909      0897 2
: 910      0898 2 MAP
: 911      0899 2 SEM_LIST: REF BBLOCK;
: 912      0900 2
: 913      0901 2 LOCAL
: 914      0902 2 SUBRTN;
: 915      0903 2
: 916      0904 2 SELECTONEU .SEM_LIST [PST$W_DATAID] OF
: 917      0905 2
: 918      0906 2 SET
: 919      0907 2 [NMA$C_PCNO_SAD]: SUBRTN = NML$SHORANGE;
: 920      0908 2 [NMA$C_PCNO_ALI]: SUBRTN = NML$SHONODEID;
: 921      0909 2 [OTHERWISE]: SUBRTN = NML$SHOPARAM;
: 922      0910 2 TES;
: 923      0911 2
: 924      0912 2 ! Call the show parameter routine using the executor data descriptor.
: 925      0913 2
: 926      0914 2 (.SUBRTN) (.SEM_LIST,
: 927      0915 2 .BUFDSC,
: 928      0916 2 .MSGSIZE,
: 929      0917 2 NML$GQ_EXEDATDSC,
: 930      0918 2 NML$GL_EXEDATPTR);
: 931      0919 2
: 932      0920 2 RETURN NML$_STS_SUC
: 933      0921 2
: 934      0922 1 END; ! End of NML$SHOEXEPARAM
```

0004 00000

.ENTRY NML\$SHOEXEPARAM, Save R2

: 0873

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHOEXEPARAM Show executor parameter

B 3
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 29
(13)

	52	04	AC	D0	00002	MOVL	SEM_LIST, R2	:	0904
	50		62	3C	00006	MOVZWL	(R2), R0	:	
038F	8F		50	B1	00009	CMPW	R0, #911	:	0907
			09	12	0000E	BNEQ	1\$:	
	51	00000000V	00	9E	00010	MOVAB	NML\$SHORANGE, SUBRTN	:	
			13	11	00017	BRB	3\$:	
0AB5	8F		50	B1	00019	CMPW	R0, #2741	:	0908
			07	12	0001E	BNEQ	2\$:	
	51	FE09	CF	9E	00020	MOVAB	NML\$SHONODEID, SUBRTN	:	
			05	11	00025	BRB	3\$:	
	51	FCC5	CF	9E	00027	MOVAB	NML\$SHOPARAM, SUBRTN	:	0909
		00000000G	00	9F	0002C	PUSHAB	NML\$GL_EXEDATPTR	:	0914
		00000000G	00	9F	00032	PUSHAB	NML\$GQ_EXEDATDSC	:	
	7E	08	AC	7D	00038	MOVQ	BUFDSC, -(SP)	:	0915
			52	DD	0003C	PUSHL	R2	:	0914
	61		05	FB	0003E	CALLS	#5, (SUBRTN)	:	
	50		01	D0	00041	MOVL	#1, R0	:	0920
			04	00	00C44	RET		:	0922

; Routine Size: 69 bytes, Routine Base: \$CODE\$ + 0310

```
936 0923 1 %SBTTL 'NML$SHORANGE Show range parameter'
937 0924 1 GLOBAL ROUTINE NML$SHORANGE (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR) =
938 0925 1
939 0926 1 !++
940 0927 1 FUNCTIONAL DESCRIPTION:
941 0928 1
942 0929 1 FORMAL PARAMETERS:
943 0930 1
944 0931 1 SEM_LIST      Parameter semantic table entry address.
945 0932 1 BUFDSC       Output message buffer descriptor address.
946 0933 1 MSGSIZE     Address of current output message size.
947 0934 1 DATDSC     QIO buffer descriptor address.
948 0935 1 DATPTR      Current pointer into QIO data buffer.
949 0936 1
950 0937 1 IMPLICIT OUTPUTS:
951 0938 1
952 0939 1 ROUTINE VALUE:
953 0940 1 COMPLETION CODES:
954 0941 1
955 0942 1 Always returns success (NML$STS_SUC).
956 0943 1
957 0944 1 --
958 0945 1
959 0946 2 BEGIN
960 0947 2
961 0948 2 MAP
962 0949 2 SEM_LIST : REF BBLOCK;
963 0950 2
964 0951 2 LOCAL
965 0952 2 CM_COUNT,
966 0953 2 RANGE_BEGIN: WORD,
967 0954 2 RANGE_END: WORD,
968 0955 2 LENGTH,
969 0956 2 PTR;
970 0957 2
971 0958 2
972 0959 2 If the address value is -1 then the parameter is not set.
973 0960 2
974 0961 2 IF .(..DATPTR)<0,32> EQLU -1 THEN
975 0962 3 BEGIN
976 0963 3 .DATPTR = ..DATPTR + 4;
977 0964 3 RETURN NML$STS_PTY;
978 0965 3 END;
979 0966 2
980 0967 2 RANGE_BEGIN = .(..DATPTR)<0,16>;
981 0968 2 RANGE_END = .(..DATPTR)<16,32>;
982 0969 2 PTR = NML$T_PRMBUFFER;
983 0970 2 CM_COUNT = T;
984 0971 2
985 0972 2 CH$WCHAR A (2, PTR);
986 0973 2 PTR = CH$MOVE (2, RANGE_BEGIN, .PTR);
987 0974 2
988 0975 2 If the range beginning = range end, don't include range end.
989 0976 2
990 0977 2 IF .RANGE_BEGIN NEQ .RANGE_END THEN
991 0978 3 BEGIN
992 0979 3 CM_COUNT = .CM_COUNT + 1;
```

```

: 993      0980      3      CH$WCHAR A (2, PTR);
: 994      0981      3      PTR = CH$MOVE (2, RANGE_END, .PTR);
: 995      0982      3      END;
: 996      0983      3
: 997      0984      2      LENGTH = .PTR - NML$T_PRMBUFFER;
: 998      0985      2
: 999      0986      2      Add coded multiple subaddresses field to output message.
1000      0987      2
1001      0988      2      NML$ADDMSGPRM (.BUFDSC,
1002      0989      2          .MSGSIZE,
1003      0990      2          .SEM_LIST [PST$W_DATAID],
1004      0991      2          .SEM_LIST [PST$B_DATATYPE] OR .CM_COUNT,
1005      0992      2          .LENGTH,
1006      0993      2          NML$T_PRMBUFFER);
1007      0994      2
1008      0995      2      Increment past range value in P4 buffer.
1009      0996      2
1010      0997      2      .DATPTR = ..DATPTR + 4;
1011      0998      2
1012      0999      2
1013      1000      2      RETURN NML$_STS_SUC;
1014      1001      1      END;
                                ! end of NML$SHORANGE
```

				001C 00000	.ENTRY	NML\$SHORANGE, Save R2,R3,R4	: 0924
		54	00000000'	00 9E 00002	MOVAB	NML\$T_PRMBUFFER, R4	: 0961
		50	14	BC D0 00009	MOVL	@DATPTR, R0	
	FFFFFFF	8F		60 D1 0000D	CMPL	(R0), #-1	
				08 12 00014	BNEQ	1\$	
	14	BC		04 C0 00016	ADDL2	#4, @DATPTR	: 0963
		50		0C CE 0001A	MNEGL	#12, R0	: 0964
				04 0001D	RET		
		51	14	BC D0 0001E	1\$: MOVL	@DATPTR, R1	: 0967
		53		61 B0 00022	MOVW	(R1), RANGE_BEGIN	
50		20		10 EF 00025	EXTZV	#16, #32, (R1), R0	: 0968
		52		50 B0 0002A	MOVW	R0, RANGE_END	
		51		64 9E 0002D	MOVAB	NML\$T_PRMBUFFER, PTR	: 0969
		50		01 D0 00030	MOVL	#1, CM_COUNT	: 0970
		81		02 90 00033	MOVB	#2, (PTR)+	: 0972
		81		53 B0 00036	MOVW	RANGE_BEGIN, (PTR)+	: 0973
		52		53 B1 00039	CMPL	RANGE_BEGIN, RANGE_END	: 0977
				08 13 0003C	BEQL	2\$	
				50 D6 0003E	INCL	CM_COUNT	: 0979
		81		02 90 00040	MOVB	#2, (PTR)+	: 0980
		81		52 B0 00043	MOVW	RANGE_END, (PTR)+	: 0981
		52		64 9E 00046	2\$: MOVAB	NML\$T_PRMBUFFER, R2	: 0984
		51		52 C2 00049	SUBL2	R2, LENGTH	
				12 BB 0004C	PUSHR	#4, R1,R4	: 0992
		51	04	AC D0 0004E	MOVL	SEM_LIST, R1	: 0991
		52	03	A1 9A 00052	MOVZBL	3(RT), R2	
	7E	52		50 C9 00056	BISL3	CM_COUNT, R2, -(SP)	
		7E		61 3C 0005A	MOVZWL	(RT), -(SP)	: 0990
		7E	08	AC 7D 0005D	MOVQ	BUFDSC, -(SP)	: 0988
	00000000G	00		06 FB 00061	CALLS	#6, NML\$ADDMSGPRM	

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHORANGE Show range parameter

E 3
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NML\$SHOPRM.B32;1

Page 32
(14)

14 BC 04 C0 00068
50 01 D0 0006C
04 04 0006F

ADDL2 #4, @DATPTR
MOVL #1, R0
RET

: 0998
: 1000
: 1001

; Routine Size: 112 bytes, Routine Base: \$CODE\$ + 0355

```
: 1016 1002 1 %SBTTL 'NML$SHOCHANNELS Show channels parameter'
: 1017 1003 1 GLOBAL ROUTINE NML$SHOCHANNELS (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR) =
: 1018 1004 1
: 1019 1005 1 ++
: 1020 1006 1 FUNCTIONAL DESCRIPTION:
: 1021 1007 1 This routine is called to format the parameter for X25 Protocol DTE
: 1022 1008 1 channels in the SHOW NICE response message. It takes the string
: 1023 1009 1 returned by the ACP in the P4 buffer and reformats it into NICE in as
: 1024 1010 1 many channel pairs as were returned in the string.
: 1025 1011 1
: 1026 1012 1 FORMAL PARAMETERS:
: 1027 1013 1
: 1028 1014 1 SEM_LIST Parameter semantic table entry address.
: 1029 1015 1 BUFDSC Output message buffer descriptor address.
: 1030 1016 1 MSGSIZE Address of current output message size.
: 1031 1017 1 DATDSC QIO buffer descriptor address.
: 1032 1018 1 DATPTR Current pointer into QIO data buffer.
: 1033 1019 1
: 1034 1020 1 IMPLICIT OUTPUTS:
: 1035 1021 1
: 1036 1022 1 ROUTINE VALUE:
: 1037 1023 1 COMPLETION CODES:
: 1038 1024 1
: 1039 1025 1 Always returns success (NML$_STS_SUC).
: 1040 1026 1
: 1041 1027 1 --
: 1042 1028 1
: 1043 1029 2 BEGIN
: 1044 1030 2
: 1045 1031 2 MAP
: 1046 1032 2 SEM_LIST : REF BBLOCK;
: 1047 1033 2
: 1048 1034 2 LOCAL
: 1049 1035 2 QIO_CHAN_LEN, ! Length of channels string in QIO P4 buffer.
: 1050 1036 2 PTR;
: 1051 1037 2
: 1052 1038 2
: 1053 1039 2 If the string length is 0 then the parameter is not set.
: 1054 1040 2
: 1055 1041 2 IF (..DATPTR)<0,16> EQL 0 THEN
: 1056 1042 3 BEGIN
: 1057 1043 3 .DATPTR = ..DATPTR + 2;
: 1058 1044 3 RETURN NML$_STS_PTY;
: 1059 1045 3 END;
: 1060 1046 2
: 1061 1047 2 QIO_CHAN_LEN = (..DATPTR)<0,16>;
: 1062 1048 2 .DATPTR = ..DATPTR + 2;
: 1063 1049 2 WHILE .QIO_CHAN_LEN GTR 0 DO
: 1064 1050 3 BEGIN
: 1065 1051 3 PTR = NML$_PRMBUFFER;
: 1066 1052 3
: 1067 1053 3 Build a temporary buffer containing a channel pair. Each element
: 1068 1054 3 in the channel pair consist of a parameter type field (2) and
: 1069 1055 3 a word of parameter value.
: 1070 1056 3
: 1071 1057 3 CH$WCHAR A (2, PTR);
: 1072 1058 3 PTR = CH$MOVE (2, ..DATPTR, .PTR);
```

```

: 1073      1059      3      .DATPTR = ..DATPTR + 2;
: 1074      1060      3      CH$WCHAR A (2, PTR);
: 1075      1061      3      PTR = CH$MOVE (2, ..DATPTR, .PTR);
: 1076      1062      3      .DATPTR = ..DATPTR + 2;
: 1077      1063      3      :
: 1078      1064      3      : Add coded multiple subaddresses field to output message.
: 1079      1065      3      :
: 1080      1066      3      NML$ADDMSGPRM (.BUFDSC,
: 1081      1067      3      .MSGSIZE,
: 1082      1068      3      .SEM_LIST [PST$W_DATAID],
: 1083      1069      3      .SEM_LIST [PST$B_DATATYPE] OR 2,
: 1084      1070      3      6,
: 1085      1071      3      NML$T_PRMBUFFER);
: 1086      1072      3      :
: 1087      1073      3      :
: 1088      1074      3      : Decrement count of channel pairs left in QIO buffer.
: 1089      1075      3      :
: 1090      1076      3      QIO_CHAN_LEN = .QIO_CHAN_LEN - 4;
: 1091      1077      3      END;
: 1092      1078      3      :
: 1093      1079      2      RETURN NML$_STS_SUC;
: 1094      1080      1      END;
                                ! end of NML$SHOCHANNELS
```

			007C 00000	.ENTRY	NML\$SHOCHANNELS, Save R2,R3,R4,R5,R6	1003
56	00000000'	00	9E 00002	MOVAB	NML\$T_PRMBUFFER, R6	
52	14	AC	D0 00009	MOVL	DATPTR, R2	1041
	00	B2	B5 0000D	TSTW	@0(R2)	
		07	12 00010	BNEQ	1\$	
62		02	C0 00012	ADDL2	#2, (R2)	1043
50		0C	CE 00015	MNEGL	#12, R0	1044
		04	04 00018	RET		
55	00	B2	3C 00019 1\$:	MOVZWL	@0(R2), QIO_CHAN_LEN	1047
62		02	C0 0001D	ADDL2	#2, (R2)	1048
54	04	AC	D0 00020	MOVL	SEM_LIST, R4	1069
		55	D5 00024 2\$:	TSTL	QIO_CHAN_LEN	1049
		37	15 00026	BLEQ	3\$	
53		66	9E 00028	MOVAB	NML\$T_PRMBUFFER, PTR	1051
83		02	90 0002B	MOVB	#2, (PTR)+	1057
83	00	B2	B0 0002E	MOVW	@0(R2), (PTR)+	1058
62		02	C0 00032	ADDL2	#2, (R2)	1059
83		02	90 00035	MOVB	#2, (PTR)+	1060
83	00	B2	B0 00038	MOVW	@0(R2), (PTR)+	1061
62		02	C0 0003C	ADDL2	#2, (R2)	1062
		56	DD 0003F	PUSHL	R6	1066
		06	DD 00041	PUSHL	#6	
50	03	A4	9A 00043	MOVZBL	3(R4), R0	1069
7E		02	C9 00047	BISL3	#2, R0, -(SP)	
7E	04	BC	3C 0004B	MOVZWL	@SEM_LIST, -(SP)	1068
7E	08	AC	7D 0004F	MOVQ	BUFDSC, -(SP)	1066
00000000G	00	06	FB 00053	CALLS	#6, NML\$ADDMSGPRM	
55		04	C2 0005A	SUBL2	#4, QIO_CHAN_LEN	1076
		C5	11 0005D	BRB	2\$	1049
50		01	D0 0005F 3\$:	MOVL	#1, R0	1079

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHOCHANNELS Show channels parameter

H 3
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 35
(15)

04 00062

RET

; 1080

; Routine Size: 99 bytes, Routine Base: \$CODE\$ + 03C5

```
1096 1081 1 %SBTTL 'NML$SHOPWSET Show password set indication'
1097 1082 1 GLOBAL ROUTINE NML$SHOPWSET (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR) =
1098 1083 1
1099 1084 1 !++
1100 1085 1 FUNCTIONAL DESCRIPTION:
1101 1086 1 This routine is called while processing a SHOW X25-SERVER DESTINATION
1102 1087 1 command ~ after the PSI ACP returns the destination's parameters.
1103 1088 1 If the password is set, it puts a password value of zero in the
1104 1089 1 NICE response message. If the password is not set, it does not
1105 1090 1 add anything to the response message.
1106 1091 1
1107 1092 1 FORMAL PARAMETERS:
1108 1093 1
1109 1094 1 SEM_LIST Parameter semantic table entry address.
1110 1095 1 BUFDSC Output message buffer descriptor address.
1111 1096 1 MSGSIZE Address of current output message size.
1112 1097 1 DATDSC QIO buffer descriptor address.
1113 1098 1 DATPTR Current pointer into QIO data buffer.
1114 1099 1
1115 1100 1 IMPLICIT OUTPUTS:
1116 1101 1
1117 1102 1 ROUTINE VALUE:
1118 1103 1 COMPLETION CODES:
1119 1104 1
1120 1105 1 Always returns success (NML$_STS_SUC).
1121 1106 1
1122 1107 1 --
1123 1108 1
1124 1109 2 BEGIN
1125 1110 2
1126 1111 2 MAP
1127 1112 2 SEM_LIST : REF BBLOCK;
1128 1113 2
1129 1114 2 LOCAL
1130 1115 2 PASSWORD_LEN;
1131 1116 2
1132 1117 2 PASSWORD_LEN = ..DATPTR<0,16>;
1133 1118 2 IF .PASSWORD_LEN GTR 0 THEN
1134 1119 2 BEGIN
1135 1120 2
1136 1121 2 Add password to message with a value of 0. This indicates simply that
1137 1122 2 the password is set, without actually returning the password.
1138 1123 2
1139 1124 2 NML$ADDMSGPRM (.BUFDSC,
1140 1125 2 .MSGSIZE,
1141 1126 2 .SEM_LIST [PST$W_DATAID],
1142 1127 2 .SEM_LIST [PST$B_DATATYPE],
1143 1128 2 1,
1144 1129 2 UPLIT (0));
1145 1130 2
1146 1131 2 END;
1147 1132 2
1148 1133 2 Increment past the password in the buffer.
1149 1134 2 .DATPTR = ..DATPTR + .PASSWORD_LEN + 2;
1150 1135 2
1151 1136 2 RETURN NML$_STS_SUC;
1152 1137 1 END; ! end of NML$SHOPWSET
```

```

                                .PSECT $SPLITS$,NOWRT,NOEXE,2
                                00000000 00008 P.AAB: .LONG 0
                                ;

                                .PSECT $CODE$,NOWRT,2
                                .ENTRY NML$SHOPWSET, Save R2,R3,R4
                                MOVL DATPTR, R4
                                MOVL (R4), R2
                                MOVZWL (R2), PASSWORD_LEN
                                BLEQ 1$
                                PUSHAB P.AAB
                                PUSHL #1
                                MOVL SEM_LIST, R0
                                MOVZBL 3(R0), -(SP)
                                MOVZWL (R0), -(SP)
                                MOVQ BUFDSC, -(SP)
                                CALLS #6, NML$ADDMSGPRM
                                MOVAB 2(PASSWORD_LEN)[R2], (R4)
                                MOVL #1, R0
                                RET
                                ; 1082
                                ; 1117
                                ; 1118
                                ; 1129
                                ; 1124
                                ; 1127
                                ; 1126
                                ; 1124
                                ; 1134
                                ; 1136
                                ; 1137

54      14      AC      D0      00002
52      64      D0      00006
53      62      3C      00009
                                1E      15      0000C
                                00      9F      0000E
00000000' 01      DD      00014
50      04      AC      D0      00016
7E      03      A0      9A      0001A
7E      60      3C      0001E
7E      08      AC      7D      00021
00000000G 00      06      FB      00025
64      02      A342    9E      0002C 1$:
50      01      D0      00031
                                04      00034

```

; Routine Size: 53 bytes, Routine Base: \$CODE\$ + 0428

```
1154 1138 1 %SBTTL 'NML$SHOCOUNTERS Show entity counters'
1155 1139 1 GLOBAL ROUTINE NML$SHOCOUNTERS (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR) =
1156 1140 1
1157 1141 1 !++
1158 1142 1 FUNCTIONAL DESCRIPTION:
1159 1143 1
1160 1144 1 This routine puts counter parameters into the response message.
1161 1145 1 Since NETACP formats the counters in NICE format, and returns them
1162 1146 1 as a string, this simply involves moving the string into the
1163 1147 1 response message with no parameter type or string length.
1164 1148 1
1165 1149 1 FORMAL PARAMETERS:
1166 1150 1
1167 1151 1 SEM_LIST Parameter semantic table entry address.
1168 1152 1 BUFDSC Output message buffer descriptor address.
1169 1153 1 MSGSIZE Address of current output message size.
1170 1154 1 DATDSC QIO buffer descriptor address.
1171 1155 1 DATPTR Current pointer into QIO data buffer.
1172 1156 1
1173 1157 1 IMPLICIT OUTPUTS:
1174 1158 1
1175 1159 1 Message buffer contains counter parameters.
1176 1160 1
1177 1161 1 ROUTINE VALUE:
1178 1162 1 COMPLETION CODES:
1179 1163 1
1180 1164 1 Always returns success (NML$_STS_SUC).
1181 1165 1
1182 1166 1 !--
1183 1167 1
1184 1168 2 BEGIN
1185 1169 2
1186 1170 2 MAP
1187 1171 2 SEM_LIST : REF BLOCK [, BYTE];
1188 1172 2
1189 1173 2 LOCAL
1190 1174 2 LEN;
1191 1175 2
1192 1176 2 LEN = ..DATPTR < 0, 16 >;
1193 1177 2 ..DATPTR = ..DATPTR + 2;
1194 1178 2
1195 1179 2 If the length is zero then no counters were returned.
1196 1180 2
1197 1181 2 IF .LEN EQL 0
1198 1182 2 THEN
1199 1183 2 RETURN NML$_STS_SUC;
1200 1184 2
1201 1185 2 NML$ADDMSGCOU ( .BUFDSC,
1202 1186 2 .MSGSIZE,
1203 1187 2 .LEN,
1204 1188 2 ..DATPTR);
1205 1189 2
1206 1190 2 ..DATPTR = ..DATPTR + .LEN;
1207 1191 2
1208 1192 2 RETURN NML$_STS_SUC
1209 1193 2
1210 1194 1 END; ! End of NML$SHOCOUNTERS
```

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine
NML\$SHOCOUNTERS Show entity counters

L 3
16-Sep-1984 00:33:36
14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 39
(17)

				0004 00000	.ENTRY NML\$SHOCOUNTERS, Save R2	:	1139
	50	14	BC	D0 00002	MOVL @DATPTR, R0	:	1176
	52		60	3C 00006	MOVZWL (R0), LEN	:	
14	BC		02	C0 00009	ADDL2 #2, @DATPTR	:	1177
			52	D5 0000D	TSTL LEN	:	1181
			14	13 0000F	BEQL 1\$:	
		14	BC	DD 00011	PUSHL @DATPTR	:	1188
			52	DD 00014	PUSHL LEN	:	1187
	7E	08	AC	7D 00016	MOVQ BUFDSC, -(SP)	:	1185
00000000G	00		04	FB 0001A	CALLS #4, NML\$ADDMSGCOU	:	
14	BC		52	C0 00021	ADDL2 LEN, @DATPTR	:	1190
	50		01	D0 00025	MOVL #1, R0	:	1192
			04	00028	RET	:	1194

; Routine Size: 41 bytes, Routine Base: \$CODE\$ + 045D

```
1212 1195 1 %SBTTL 'NML$SHOOWNER Translate Data Link Mapping bit to Owner'
1213 1196 1 GLOBAL ROUTINE NML$SHOOWNER (SEM_LIST, BUFDSC, MSGSIZE, DATDSC, DATPTR)=
1214 1197 1
1215 1198 1 !++
1216 1199 1 FUNCTIONAL DESCRIPTION:
1217 1200 1
1218 1201 1 This routine is called when doing a SHOW CIRC CHAR. It looks
1219 1202 1 at the bit value returned by the ACP for DLM (Data Link Mapping),
1220 1203 1 and, if it's set, returns an OWNER parameter value for the
1221 1204 1 executor node to NCP. The executor node is the only value
1222 1205 1 currently allowed for OWNER.
1223 1206 1
1224 1207 1
1225 1208 1 FORMAL PARAMETERS:
1226 1209 1
1227 1210 1 SEM_LIST Parameter semantic table entry address.
1228 1211 1 BUFDSC Output message buffer descriptor address.
1229 1212 1 MSGSIZE Address of current output message size.
1230 1213 1 DATDSC QIO buffer descriptor address.
1231 1214 1 DATPTR Current pointer into QIO data buffer.
1232 1215 1
1233 1216 1 ROUTINE VALUE:
1234 1217 1 COMPLETION CODES:
1235 1218 1
1236 1219 1 Always returns success (NML$_STS_SUC).
1237 1220 1
1238 1221 1 --
1239 1222 1
1240 1223 2 BEGIN
1241 1224 2
1242 1225 2 MAP
1243 1226 2 SEM_LIST : REF BLOCK [, BYTE];
1244 1227 2
1245 1228 2 BIND EXECUTOR = UPLIT BYTE
1246 1229 2 (NMA$M_PTY_COD+1, NMA$C_ENT_NOD, ! Entity type = node
1247 1230 2 2, WORD (0)); ! Node address = 0 (executor)
1248 1231 2
1249 1232 2 .DATPTR = ..DATPTR + 4;
1250 1233 2
1251 1234 2 ! If the address value is -1 then the owner is not set.
1252 1235 2 ! If the bit value is clear, then there is no owner specified.
1253 1236 2
1254 1237 2 IF (..DATPTR - 4) < 0, 32> EQLU -1 OR
1255 1238 3 NOT (..DATPTR - 4) < 0, 32>)
1256 1239 2 THEN
1257 1240 2 RETURN NML$_STS_PTY;
1258 1241 2
1259 1242 2 !
1260 1243 2 ! Add coded multiple executor node id field to output message.
1261 1244 2
1262 1245 2 NML$ADDMSGPRM (.BUFDSC,
1263 1246 2 .MSGSIZE,
1264 1247 2 .SEM_LIST [PST$W_DATAID],
1265 1248 2 .SEM_LIST [PST$B_DATATYPE] OR 2,
1266 1249 2 5,
1267 1250 2 EXECUTOR);
1268 1251 2
```

NML\$SHOPRM
V04-000

N 3
NML special volatile parameter handling routine 16-Sep-1984 00:33:36
NML\$SHOOWNER Translate Data Link Mapping bit t 14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 41
(18)

: 1269
: 1270

1252 2 RETURN NML\$_STS_SUC
1253 1 END;

! End of NML\$SHOOWNER

```
.PSECT $PLITS$,NOWRT,NOEXE,2
02 00 81 0000C P.AAC: .BYTE -127, 0, 2
      0000 0000F .WORD 0
EXECUTOR= P.AAC

.PSECT $CODES$,NOWRT,2
.ENTRY NML$SHOOWNER, Save nothing
ADDL2 #4, @DATPTR
MOVL @DATPTR, R0
CMPL -4(R0), #-1
BEQL 1$
BLBS -4(R0), 2$
MNEGL #12, R0
RET
PUSHAB EXECUTOR
PUSHL #5
MOVL SEM_LIST, R0
MOVZBL 3(R0), R1
BISL3 #2, R1, -(SP)
MOVZWL (R0), -(SP)
MOVQ BUFDSC, -(SP)
CALLS #6, NML$ADDMSGPRM
MOVL #1, R0
RET
```

; Routine Size: 66 bytes, Routine Base: \$CODES + 0486

NML\$SHOPRM
V04-000

NML special volatile parameter handling routine 16-Sep-1984 00:33:36
NML\$SHOOWNER Translate Data Link Mapping bit t 14-Sep-1984 12:50:20

VAX-11 Bliss-32 V4.0-742
[NML.SRC]NMLSHOPRM.B32;1

Page 42
(19)

: 1272 1254 1 END
: 1273 1255 1
: 1274 1256 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	256	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$SPLITS	17	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	1224	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[NML.OBJ]NMLLIB.L32;1	341	22	6	27	00:00.1
-\$255\$DUA28:[SHRLIB]NMLIBRY.L32;1	887	15	1	47	00:00.2
-\$255\$DUA28:[SHRLIB]NET.L32;1	1279	3	0	63	00:00.3
-\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	0	0	581	00:03.3

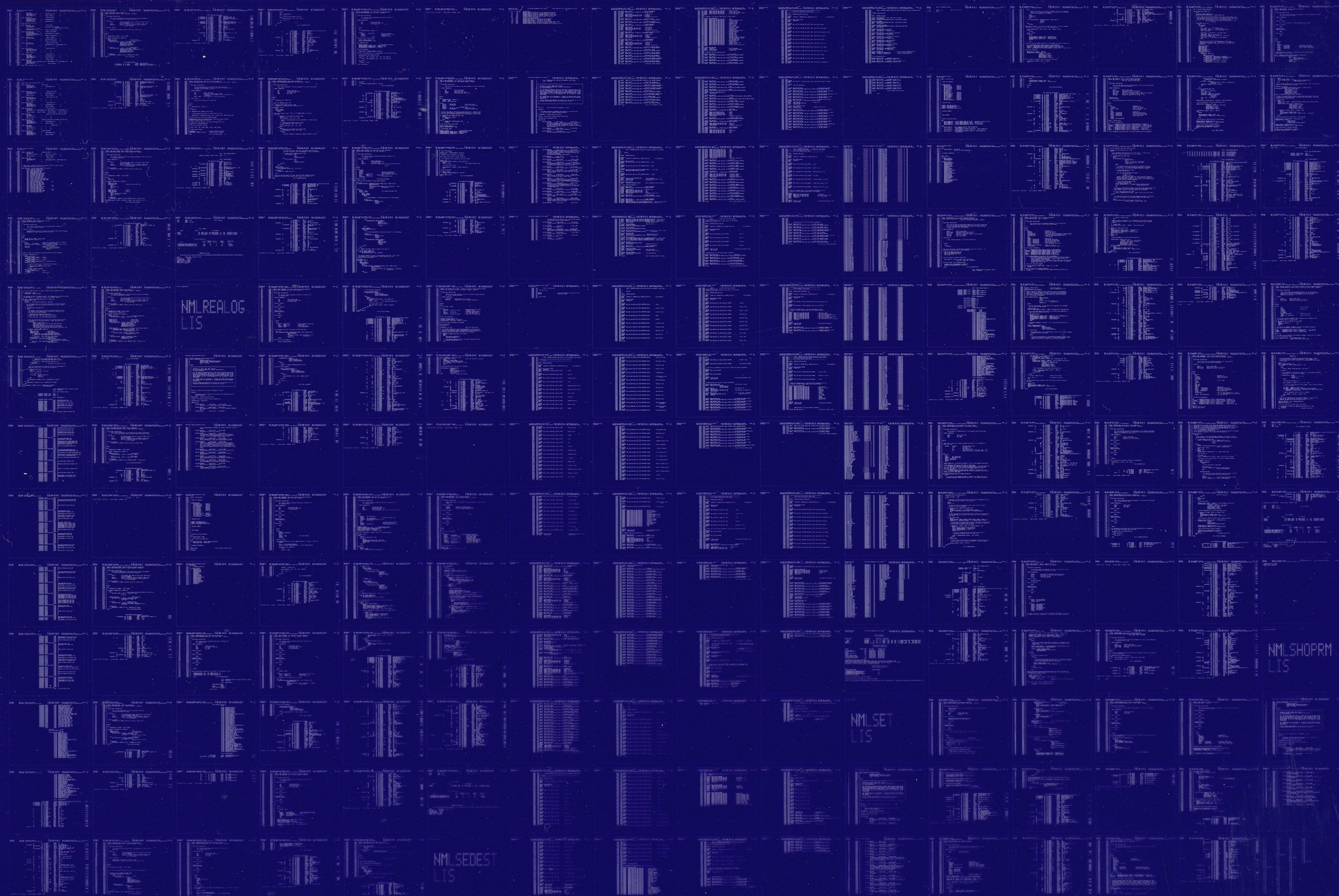
COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:NMLSHOPRM/OBJ=OBJ\$:NMLSHOPRM MSRC\$:NMLSHOPRM/UPDATE=(ENH\$:NMLSHOPRM)

: Size: 1224 code + 273 data bytes
: Run Time: 00:27.7
: Elapsed Time: 01:04.7
: Lines/CPU Min: 2721
: Lexemes/CPU-Min: 10955
: Memory Used: 128 pages
: Compilation Complete

0286 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0287 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

